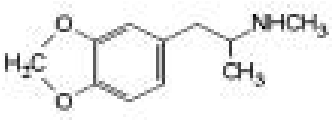
MaDMAn_H3rCuL3s

nTitles Protect REViSiTED

Version 1.0
November 2006



1. Forewords

It seems that either nTitles has either updated their protection or just I came across one that is a special version. Knowing what we learned from the last tutorial, this one should seem very familiar. There is only a few changes, but the protection basically remains the same.

BEE Seeing YA.

MDMA

MDMA

Editor: MaDMAn_H3rCuL3s

Disclaimers

All code included with this tutorial is free to use and modify; we only ask that you mention where you found it. This tutorial is also free to distribute in its current unaltered form, with all the included supplements.

All the commercial programs used within this document have been used only for the purpose of demonstrating the theories and methods described. No distribution of patched applications has been done under any media or host. The applications used were most of the times already been patched, and cracked versions were available since a lot of time. ARTeam or the authors of the paper cannot be considered responsible damages the companies holding rights on those programs. The scope of this tutorial as well as any other ARTeam tutorial is of sharing knowledge and teaching how to patch applications, how to bypass protections and generally speaking how to improve the RCE art. We are not releasing any cracked application.

Verification

ARTeam.esfv can be opened in the ARTeamESFVChecker to verify all files have been released by ARTeam and are unaltered. The ARTeamESFVChecker can be obtained in the release section of the ARTeam site: http://releases.accessroot.com

Table of Contents

1. .nTitles Protect REViSiTED

1.1. Abstract

After a full reading you should be able to unpack anything nTitles has to throw at you. Assuming the protection did change, we will probably see a new variant floating around soon, following the release of this tutorial.

1.2. Targets

Applications are updated at a regular interval, given that, the target used in this tutorial will be available from the link provided.

No Hassle File Transfer v1.0
http://arteam.accessroot.com/tools/NHFT_Setup.zip

1.3. Removing the Protection

1.3.1 Preparation

You will need the following tools to proceed:
1. OllyDBG
2. Lord-PE

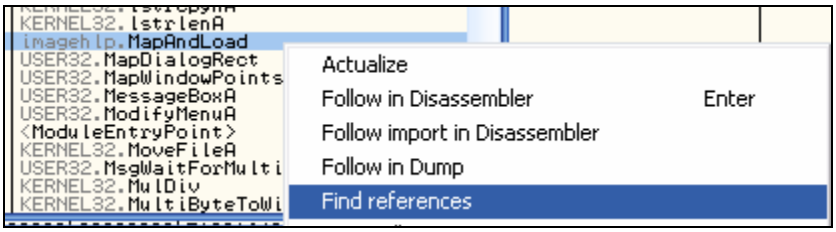
1.3.2 Checking out the target

Just when we thought this chapter was written, it seems nTitles has brought a new version or maybe even a version I have not seen before. Like in the last tutorial we will assume that you will register for a trial registration key. This way we can bypass the registration checks and go straight to unpacking. Open up exe in Olly and you should see nTitles EP:

```
Address   Hex dump          Disassembly
00477AC8  $  6A 60          PUSH 60
00477ACA  .  68 00805000    PUSH No_Hassl.00508000
00477ACF  .  E8 744D0000    CALL No_Hassl.0047C848
00477AD4  .  BF 94000000    MOV EDI,94
00477AD9  .  8BC7           MOV EAX,EDI
00477ADB  .  E8 70FEFFFF    CALL No_Hassl.00477950
00477AE0  .  8965 E8        MOV DWORD PTR SS:[EBP-18],ESP
00477AE3  .  8BF4           MOV ESI,ESP
00477AE5  .  893E           MOV DWORD PTR DS:[ESI],EDI
00477AE7  .  56             PUSH ESI
00477AE8  .  FF15 78934F00  CALL DWORD PTR DS:[<&KERNEL32.GetVersionExA>]
00477AEE  .  8B4E 10        MOV ECX,DWORD PTR DS:[ESI+10]
00477AF1  .  890D CC465400  MOV DWORD PTR DS:[5446CC],ECX
00477AF7  .  8B46 04        MOV EAX,DWORD PTR DS:[ESI+4]
00477AFA  .  A3 D8465400    MOV DWORD PTR DS:[5446D8],EAX
00477AFF  .  8B56 08        MOV EDX,DWORD PTR DS:[ESI+8]
00477B02  .  8915 DC465400  MOV DWORD PTR DS:[5446DC],EDX
00477B08  .  8B76 0C        MOV ESI,DWORD PTR DS:[ESI+C]
00477B0B  .  81E6 FF7F0000  AND ESI,7FFF
00477B11  .  8935 D0465400  MOV DWORD PTR DS:[5446D0],ESI
00477B17  .  83F9 02        CMP ECX,2
00477B1A  .v 74 0C          JE SHORT No_Hassl.00477B28
```

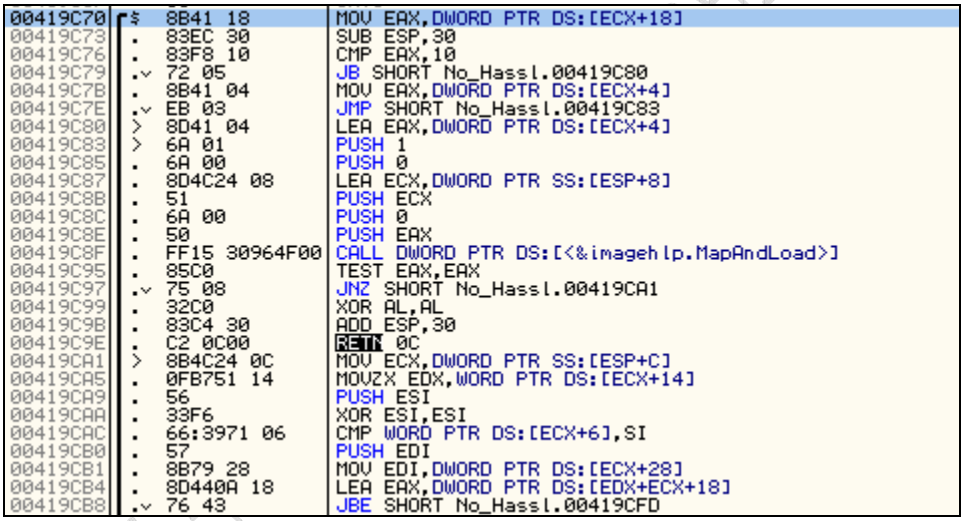
Entrypoint.

Now we need to understand that my older method of the ImageLoad/ImageUnload still exists, but doesn't actually help us too much. The only thing it actually does is get us real close to where we want to be. So hit ALT+N and then let's look for our API's.



Our API.

Follow in Disassembler.



Same code from before.

So let's set a BP after the JNZ, so our image is created, and we can see what it looks like. So on line 0x00419CA1 set a BP and run it (also bypass the "I wanna try it")

```
Stack SS:[0121FC0C]=005F0080, (ASCII "PE")
ECX=0000D67E
Jump from 00419C97
```

Address	Hex dump	ASCII
005F0000	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00	MZÉ.♥...♦... ..
005F0010	B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00	╕.......@.......
005F0020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	....
005F0030	00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 00	............Ç...
005F0040	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	♫▼‖♫.┤.=╛┐0L=!Th
005F0050	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno
005F0060	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS
005F0070	6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00	mode....$.......
005F0080	50 45 00 00 4C 01 04 00 A5 1B 49 45 00 00 00 00	PE..L0♦.ñ+IE....
005F0090	00 00 00 00 E0 00 0E 01 0B 01 08 00 00 A0 01 00	....∝.ß0∂0◙..à0.
005F00A0	00 40 00 00 00 00 00 00 8E B4 01 00 00 20 00 00	.@......Ä┤0.. ..
005F00B0	00 C0 01 00 00 00 40 00 00 20 00 00 00 10 00 00	.└0...@.. ...►..
005F00C0	04 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00	♦.......♦.......
005F00D0	00 20 02 00 00 10 00 00 00 00 00 00 02 00 00 04	. ☻..►......☻..♦
005F00E0	00 00 10 00 00 10 00 00 00 00 10 00 00 10 00 00	..►..►....►..►..
005F00F0	00 00 00 00 10 00 00 00 00 00 00 00 00 00 00 00	....►...........
005F0100	40 B4 01 00 4B 00 00 00 00 E0 01 00 A8 1E 00 00	@┤0.K....∝0.¿▲..
005F0110	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	....
005F0120	00 00 02 00 0C 00 00 00 00 C0 01 00 1C 00 00 00	..☻.......└0.└...
005F0130	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	....
005F0140	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	....
005F0150	00 00 00 00 00 00 00 00 00 20 00 00 08 00 00 00	....◘...
005F0160	00 00 00 00 00 00 00 00 08 20 00 00 48 00 00 00	....H...
005F0170	00 00 00 00 00 00 00 00 2E 74 65 78 74 00 00 00	.........text...
005F0180	94 94 01 00 00 20 00 00 00 A0 01 00 00 10 00 00	öö0.. ...à0..►..
005F0190	00 00 00 00 00 00 00 00 00 00 00 00 20 00 00 60	....`

Image

This really doesn't help us aside from bringing us real close to where we need to be. So hit CTRL+F9 and land on Return. When you do land on Return you will notice the image is gone. We don't care. Hit F8 and get out of this function, and trace till this:

```
0044CB25  .  83C4 04        ADD ESP,4
0044CB28  .  51             PUSH ECX                          PUSH SIZE
0044CB29  .  50             PUSH EAX                          PUSH LOCATION
0044CB2A  .  8D4C24 5C      LEA ECX,DWORD PTR SS:[ESP+5C]
```

We see our New Image about to be created.

```
EAX 00F211A0
```

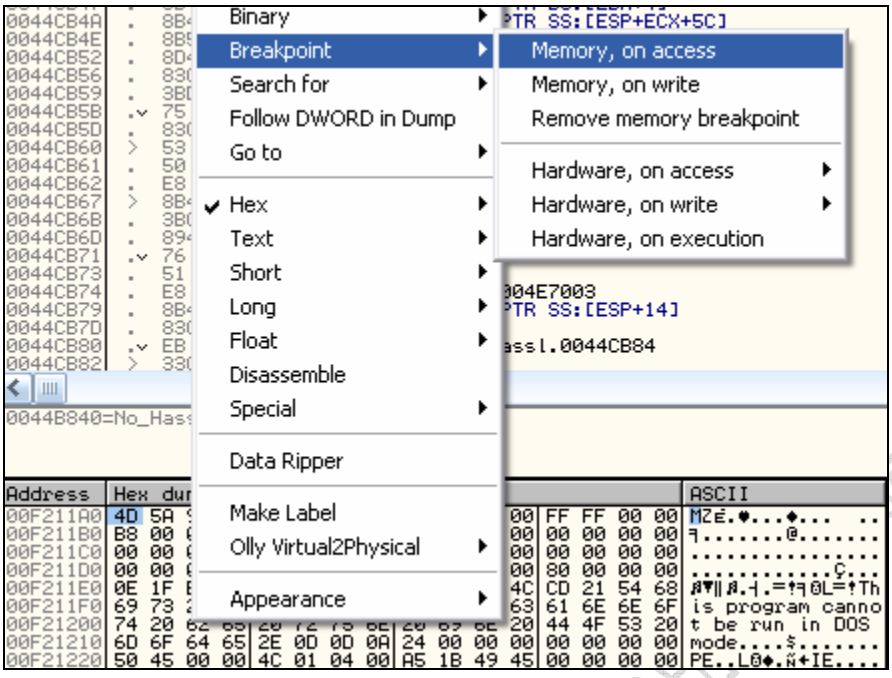
Location

```
ECX 0001F000
```

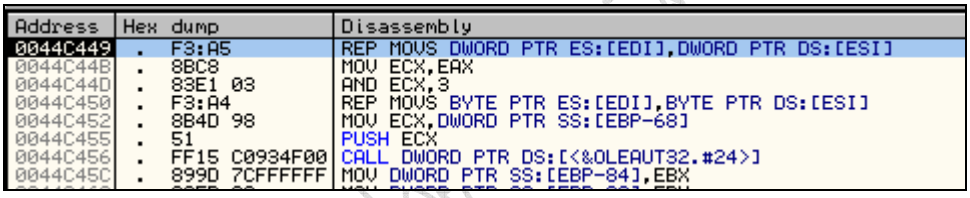
Size

Address	Hex dump	ASCII
00F211A0	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00	MZÉ.♥...♦... ..
00F211B0	B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00	╕.......@.......
00F211C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	....
00F211D0	00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 00	............Ç...
00F211E0	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	♫▼‖♫.┤.=╛┐0L=!Th
00F211F0	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno
00F21200	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS
00F21210	6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00	mode....$.......
00F21220	50 45 00 00 4C 01 04 00 A5 1B 49 45 00 00 00 00	PE..L0♦.ñ+IE....
00F21230	00 00 00 00 E0 00 0E 01 0B 01 08 00 00 A0 01 00	....∝.ß0∂0◙..à0.
00F21240	00 40 00 00 00 00 00 00 8E B4 01 00 00 20 00 00	.@......Ä┤0.. ..
00F21250	00 C0 01 00 00 00 40 00 00 20 00 00 00 10 00 00	.└0...@.. ...►..
00F21260	04 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00	♦.......♦.......
00F21270	00 20 02 00 00 10 00 00 00 00 00 00 02 00 00 04	. ☻..►......☻..♦
00F21280	00 00 10 00 00 10 00 00 00 00 10 00 00 10 00 00	..►..►....►..►..
00F21290	00 00 00 00 10 00 00 00 00 00 00 00 00 00 00 00	....►...........
00F212A0	40 B4 01 00 4B 00 00 00 00 E0 01 00 A8 1E 00 00	@┤0.K....∝0.¿▲..
00F212B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	....
00F212C0	00 00 02 00 0C 00 00 00 00 C0 01 00 1C 00 00 00	..☻.......└0.└...
00F212D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	....
00F212E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	....
00F212F0	00 00 00 00 00 00 00 00 00 20 00 00 08 00 00 00	....◘...
00F21300	00 00 00 00 00 00 00 00 08 20 00 00 48 00 00 00	....H...
00F21310	00 00 00 00 00 00 00 00 2E 74 65 78 74 00 00 00	.........text...
00F21320	94 94 01 00 00 20 00 00 00 A0 01 00 00 10 00 00	öö0.. ...à0..►..
00F21330	00 00 00 00 00 00 00 00 00 00 00 00 20 00 00 60	....`

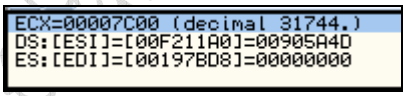
Our new image!



Set the memory Breakpoint.



We land here.



Where we write.

So we see our image is written from [ESI[to [EDI], which is very similar to my last tutorial with the exception that upon first load of image we can place breakpoints and trace it that way. Here we must first let it load image then delete it, then remap it again and from then on we can trace it. So let's follow the image in dump and see it: